SE 461 Computer Vision

Nazar Khan
PUCIT
Lecture 4

Convolution

- Convolution implies
  - Spatial filtering
- What is a filter?
  - Something that lets some things pass through and prevents the rest from passing through.
    - Oil filter, Air filter, Noise filter

Convolution

- We have seen in Lecture 2 that convolution with an averaging mask yields
  - a smooth version of the input signal
  - by suppressing sharp changes (noise)
- The mask is also called a **filter**.
  - Why?
- Accordingly, convolution is also called **filtering**.
- Convolution with other masks/filters can yield different results
  - Derivative filtering for edge detection.

Applying Masks to Images

- Convolution Operation
- Mask
  - Set of pixel positions and weights
  - Origin of mask

1	1	1
1	**1**	1
1	1	1

1	2	1
2	**4**	2
1	2	1

1
1
1
1
1

Applying Masks to Images

- $I_1 \otimes$ mask $= I_2$
- Convention: I_2 is the same size as I_1
- Mask Application:
  - For each pixel
  - Place mask origin on top of pixel
  - Multiply each weight with pixel under it
  - Sum the result and put in location of the pixel

discrete convolution in 1-D:

$$(\boldsymbol{f} * \boldsymbol{w})_i \;:=\; \sum_{k=-\infty}^{\infty} f_{i-k}\, w_k$$

discrete convolution in 2-D:

$$(\boldsymbol{f} * \boldsymbol{w})_{i,j} \;:=\; \sum_{k=-\infty}^{\infty} \sum_{\ell=-\infty}^{\infty} f_{i-k,\,j-\ell}\, w_{k,\ell}$$

Applying Masks to Images

40	40	40	80	80	80
40	40	40	80	80	80
40	**1/9** 40	**1/9** 40	**1/9** 80	80	80
40	**1/9** 40	**1/9** 40	**1/9** 80	80	80
40	**1/9** 40	**1/9** 40	**1/9** 80	80	80
40	40	40	80	80	80

1/9 x

1	1	1
1	1	1
1	1	1

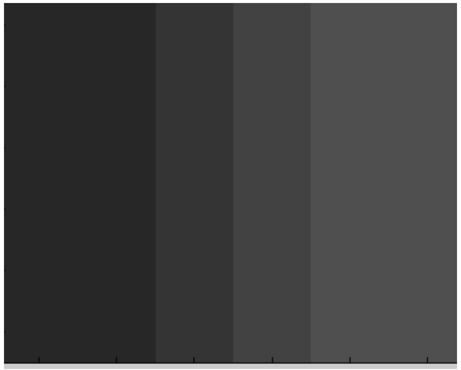
$6*(1/9*40)+3*(1/9*80) = 53$

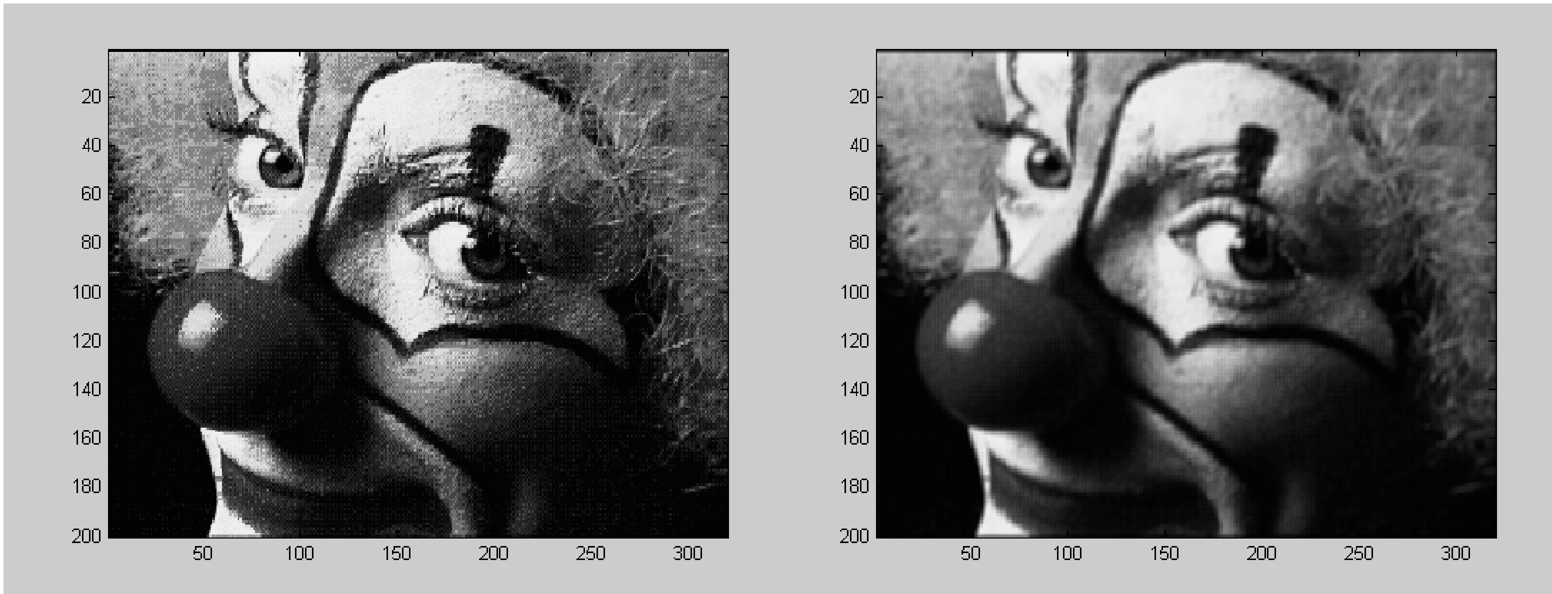
Applying Masks to Images

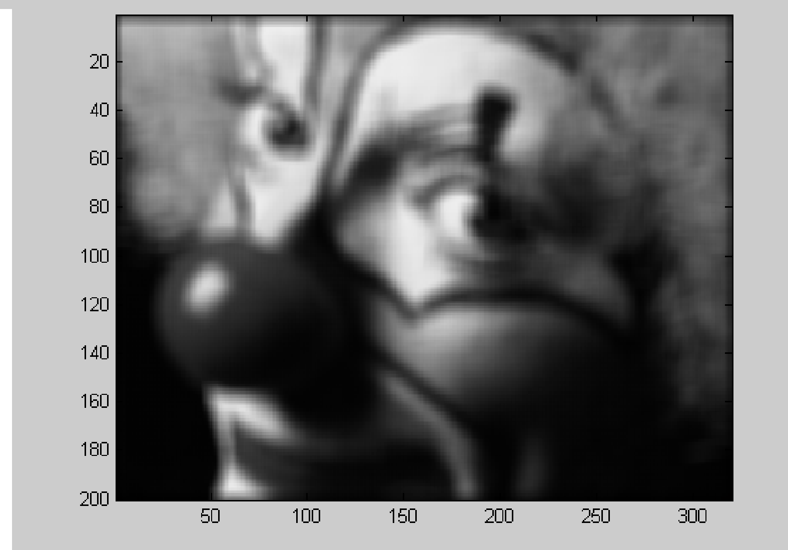
40	40	53	67	80	80
40	40	53	67	80	80
40	40	53	67	80	80
40	40	53	67	80	80
40	40	53	67	80	80
40	40	53	67	80	80

- Overall effect of this mask?

- Smoothing filter




20 40 60 80 100 120 140 160 180 200
50 100 150 200 250 300
20 40 60 80 100 120 140 160 180 200
50 100 150 200 250 300


20 40 60 80 100 120 140 160 180 200
50 100 150 200 250 300

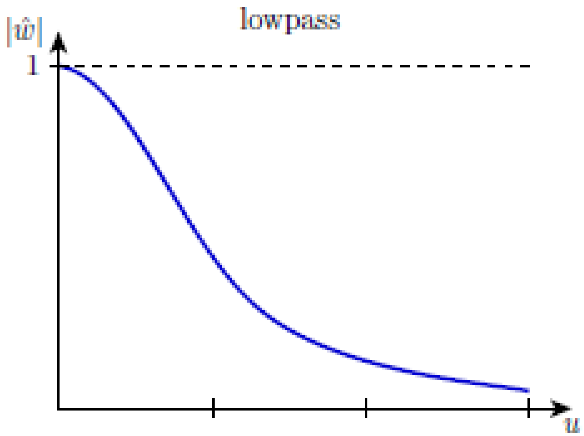
What about edge and corner pixels?

- Expand image with virtual pixels
- Options
  - Fill with a particular value, e.g. zeros
  - Fill with nearest pixel value
  - Mirrored boundary
- Fatalism: just ignore them (not recommended)

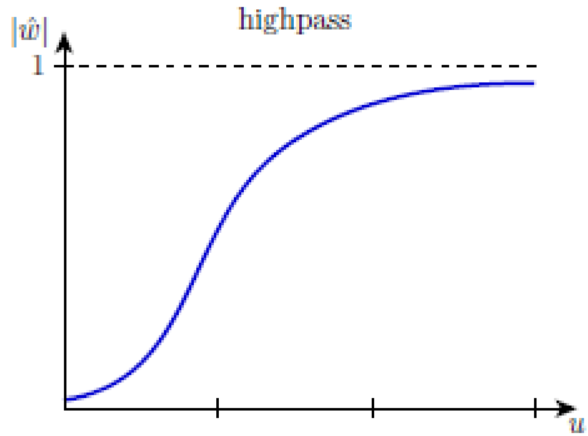
Frequency Interpretation

- Noise is the high frequency component of a signal.
- Convolution with averaging mask is equivalent to reducing the high frequency components of a signal.

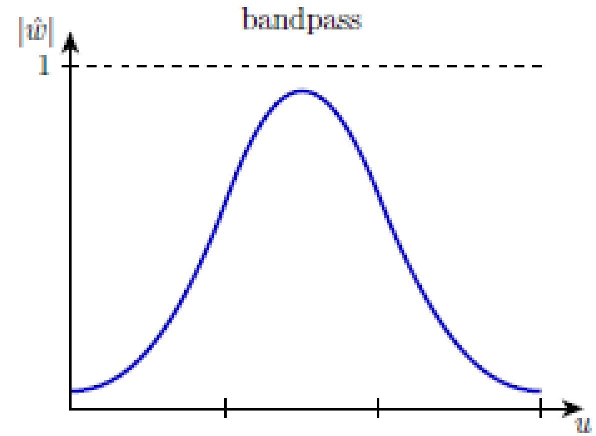
Frequency Interpretation



Lowpass:
Reduce high
frequencies.



Highpass:
Reduce low
frequencies.



Bandpass:
Reduce
frequencies
outside a
certain band.

Frequency Interpretation

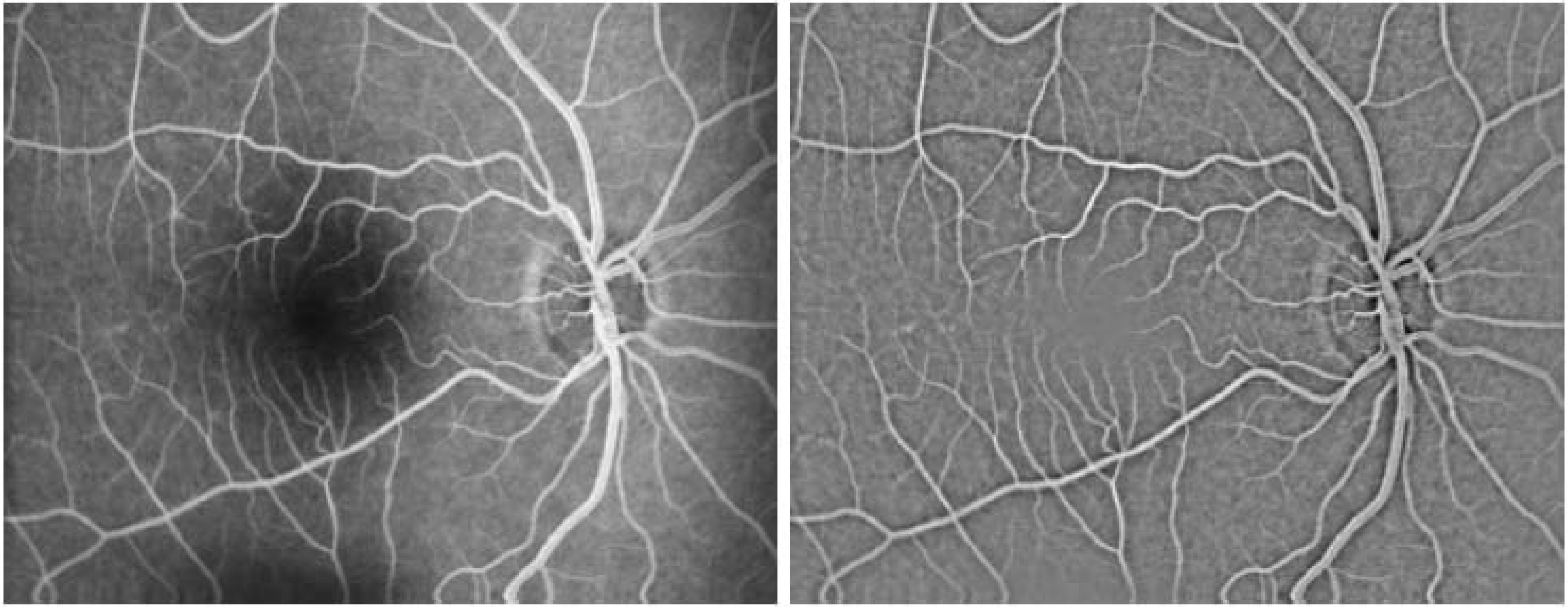
Lowpass filters: Low frequencies are less attenuated than high ones.

Highpass filters: High frequencies are less attenuated than low ones.

Bandpass filters: Structures within a specific frequency band are hardly attenuated.

Highpass Filtering

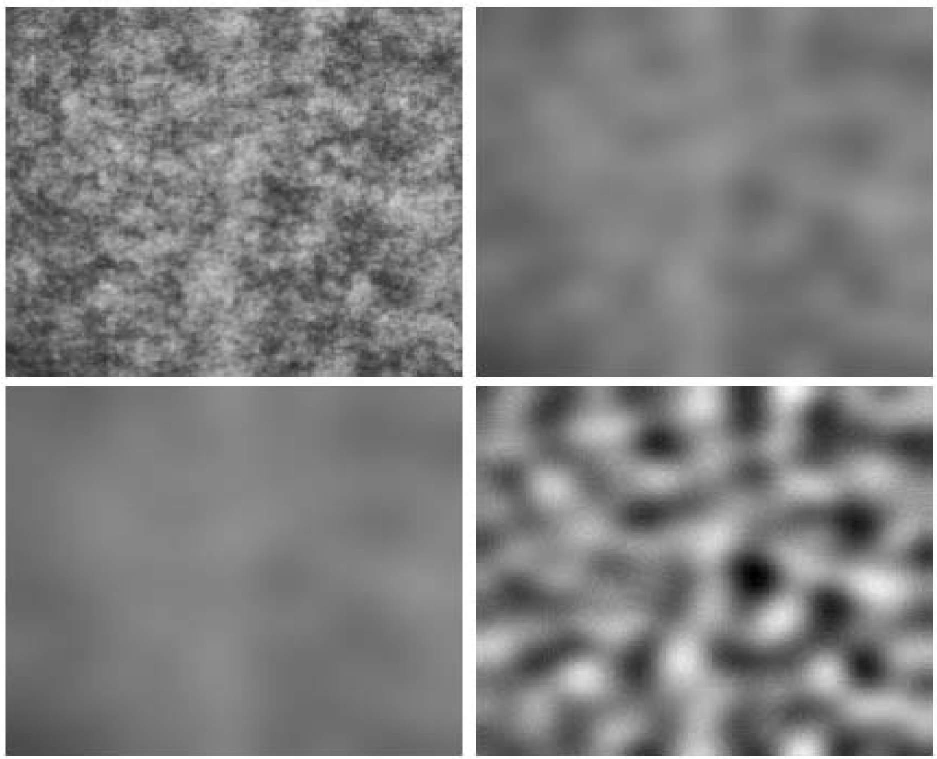
- H=I-Gaussian*I



Left Vessel structure of the background of the eye. **Right:** Elimination of low-frequent background structures by subtracting a Gaussian-smoothed version from the original image. The greyscale range $[-94, 94]$ has been rescaled to $[0, 255]$ by an affine rescaling. Author: J. Weickert (2002).

Bandpass Filtering

- $B = G1 * I - G2 * I$



(a) Top left: Fabric, 257 × 257 pixels. **(b) Top right:** After lowpass filtering with a Gaussian with $\sigma = 10$. **(c) Bottom left:** Lowpass filtering with $\sigma = 15$. **(d) Bottom right:** Subtracting (b) and (c) gives a bandpass filter that visualises cloudiness on a certain scale. The greyscale range has been affinely rescaled from $[-13, 13]$ to $[0, 255]$. Author: J. Weickert (2002).

Some Properties of Convolution

Linearity:

$$(\alpha f + \beta g) * w \;=\; \alpha\,(f * w) + \beta\,(g * w) \qquad \forall\, \alpha, \beta \in \mathbb{R}.$$

Shift Invariance:

$$T_b(f * w) = (T_b f) * w$$

for all translations T_b.

Commutativity:

$$f * w \;=\; w * f.$$

Function and convolution kernel play an equal role.

Associativity:

$$(f * v) * w \;=\; f * (v * w).$$

Successive convolution with kernels v and w comes down to a single convolution with kernel $v * w$.